How to train a linear classifier

- get a set of labelled examples

$$(\underline{x}_i, y_i)$$

$$y_i = \begin{cases} 1 \\ -1 \end{cases}$$

- Split into test, training, validation sets
- Choose a loss
- For several different values of Θ
  - compute $\hat{\omega}, \hat{b}$ to minimize

$$\Theta \|\omega\|^2 + \frac{1}{N} \sum_{i \in \text{train}} \mathcal{L}(x_i, y_i, \omega, b)$$

loss.

  - compute

$$\mathcal{L}_V = \frac{1}{N} \sum_{i \in \text{val}} \mathcal{L}(x_i, y_i, \hat{\omega}, \hat{b})$$

- Choose Θ with best $\mathcal{L}_V$
- retrain on train $\cup$ val.

How to evaluate a linear classifier (24)

- On test, compute

~~N_{FP} =~~

$$FP = \frac{\#\{\text{negative examples labelled +}\}}{\#\{\text{total tested}\}}$$

$$F_N = \frac{\#\{\text{positive egs labelled -}\}}{\#\{\text{total tested}\}}$$

OR:

$$\text{recall} = \frac{\#\{\text{+ve examples labelled +}\}}{\#\{\text{+ve examples}\}}$$

$$\text{precision} = \frac{\#\{\text{~~exa~~ +ve examples labelled +}\}}{\#\{\text{all examples labelled +}\}}$$

Q: How do we minimize this cost function?

Generally, minimization requires some idea of direction in which function goes down.

1D case:

prob: choose x to minimize $f(x)$

(assume f has a derivative)

$$f(x + \delta x) \approx f(x) + \delta x \frac{df}{dx}$$

so if we're at x_n,

$$x_{n+1} = x_n - h \frac{df}{dx}$$ is a good choice

$$f(x_{n+1}) \approx f(x_n) - h\left(\frac{df}{dx}\right)^2$$

So f goes down IF h is small enough.

Strategy: choose seq. h_i s.t.

$$\lim_{i \to \infty} h_i \Rightarrow 0$$

$$\lim_{n \to \infty} \left[\sum_{i=1}^{n} h_i\right] = \infty$$

(eg. $h_i = \frac{1}{i}$)

then

$$x_{i+1} = x_i - h_i \frac{df}{dx}$$

will converge to a min.

in higher dimensions, we must use partial derivatives

$$f(x+\delta x, y) \approx f(x,y) + \delta x \frac{\partial f}{\partial x}$$

$$f(x, y+\delta y) \approx f(x,y) + \delta y \frac{\partial f}{\partial y}$$

and

$$f(x+\delta x, y+\delta y) \approx f(x,y) + \delta x \frac{\partial f}{\partial x} + \delta y \frac{\partial f}{\partial y}$$

We can write this in a vector notation **

$$\begin{pmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{pmatrix} = \nabla f \quad ; \quad \begin{pmatrix} \delta x \\ \delta y \end{pmatrix} = \delta \underline{u}$$

$$\begin{pmatrix} x \\ y \end{pmatrix} = \underline{u}$$

$$f(\underline{u} + \delta\underline{u}) \simeq f(\underline{u}) + \delta\underline{u}^T \nabla f$$

- this works for any dimension

i.e $\underline{u} = \begin{pmatrix} x_0 \\ x_1 \\ \vdots \\ x_n \end{pmatrix}$; $\delta\underline{u} = \begin{pmatrix} \delta x_0 \\ \vdots \\ \delta x_n \end{pmatrix}$

$$\nabla f = \begin{pmatrix} \partial f/\partial x_0 \\ \vdots \\ \partial f/\partial x_n \end{pmatrix}$$

Now we can make our min alg work for arbitrary dimension

Notice

~~$f(\underline{u} - h\nabla f)$~~

$$f(\underline{u} - h\nabla f) \simeq f(\underline{u}) - h(\nabla f^T \nabla f)$$

Choose h_i a sequence s.t.

- $\lim_{i \to \infty} h_i = 0$
- $\lim_{n \to \infty} \left[\sum_{i=1}^{n} h_i \right] = \infty$

x_0 start point

$$x_{i+1} = x_i - h_i \nabla f$$

will eventually converge to a min.

mostly: we do

$$x_{i+1} = x_i - h_i \frac{\nabla f}{\| \nabla f \|}$$

- so the direction of the update is a unit vector.

Now consider a linear classifier we have

$$\Theta \|w\|^2 + \frac{1}{N} \sum_{i \in \text{train}} \mathcal{L}(x_i, y_i, w, b).$$

the parameters are w, b

Computing the gradient might be very expensive — sum over all examples